

APPLICATION FOR UNITED STATES PATENT

For

**METHOD AND APPARATUS FOR RESETABLE MEMORY AND DESIGN  
APPROACH FOR SAME**

Inventor:

Vijay K. Seshadri

Kenneth S. McElvain

Prepared by:

BLAKELY SOKOLOFF TAYLOR & ZAFMAN LLP  
32400 Wilshire Boulevard  
Los Angeles, CA 90025-1026  
(408) 720-8598

Attorney's Docket No.: 02986P013

"Express Mail" mailing label number: EL371009981US

Date of Deposit: March 4, 2002

I hereby certify that I am causing this paper or fee to be deposited with the United States Postal Service "Express Mail Post Office to Addressee" service on the date indicated above and that this paper or fee has been addressed to the Commissioner for Patents, Washington, D. C. 20231

JANECE SHANNON  
(Typed or printed name of person mailing paper or fee)

Janece Shannon  
(Signature of person mailing paper or fee)

3/4/2002  
(Date signed)

FILED IN 371009981

# METHOD AND APPARATUS FOR RESETABLE MEMORY AND DESIGN APPROACH FOR SAME

## FIELD OF THE INVENTION

**[0001]** The field of invention relates generally to electronic circuit design; and more specifically, to a method and apparatus for a resetable memory and a design approach for same.

## BACKGROUND

**[0002]** **Figure 1** shows a memory unit 101. The memory unit 101 may be viewed as having a plurality of storage cells (or simply, "cells"). Associated with each cell is a unique address that provides access to the location of a particular storage cell. Each storage cell has the capacity to store "n" bits (where n is an integer greater than or equal to one). The n bits may be collectively referred to as a word of data.

**[0003]** Often, a memory unit 101 is written to by: 1) providing a word of data (i.e., "n" bits of data as seen in **Figure 1**) to be written into the memory unit 101 on a data bus (such as input data bus 103 observed in **Figure 1**); 2) providing an address (e.g., on address bus 102) that defines which storage cell will store the word of data; and 3) presenting a signal to the memory unit that effectively indicates the word of data on the data bus is to be written into the memory unit (such activating the write enable (WE) control line 104 of **Figure 1**).

Alternatively, the write could be time based as in a shift register, or it could be a mix of address and time based write.

**[0004]** Often, a memory unit 101 is read from by: 1) providing an address (e.g., on address bus 102) that defines which cell a word of data will be read from; and 2) presenting a signal to the memory unit that effectively indicates a word of data is to be read from the memory unit ( such as activating the read enable (RE) control line 130 of **Figure 1**). The word of data is presented at the data output bus 106. In embodiments alternative to that observed in **Figure 1**, the data in and data out buses 103, 106 may be combined to form a bi-directional bus. Also, in various alternate embodiments, one of the enable lines 104, 130 may be eliminated so that a single line is used to toggle the memory unit between being in a writable state and being in a readable state. Alternatively, it could be time based as in a shift register, or a mix of address and time based read. A commercial example that uses a mix of address and time based read/write is the Xilinx Virtex SRL primitive that is written into like a shift register and is read from like a RAM.

**[0005]** The memory unit 101 of **Figure 1** can be used to implement a number of storage related devices such as a random access memory (RAM), a first-in-first-out (FIFO) queue (e.g., by appropriately controlling the address values of the memory unit 101 such that a FIFO queue is emulated with the memory unit 101), a content addressable memory (CAM), a shift register, etc. A problem with memory units (as they are offered to designers who wish to employ them in their circuit designs), however, is that they do not have a reset function. A reset function effectively “clears” the memory unit’s cell word values to some “reset” value (e.g., a n wide value of “0”); and, often, the integration of circuitry for

resetting the cell word values of the memory unit 101 is too expensive and/or complicated to implement. For example, according to one approach, in order to implement a resetable memory, each n wide storage cell is implemented with resetable flip-flops that are individually accessed via complicated multiplexing and control circuitry. Here, the use of resetable flip-flops to implement each n wide storage cell (as well as the complicated multiplexing and control circuitry) can result in a resetable memory unit having noticeably slower performance (and that consumes more silicon surface area) than a memory unit that does not have resetable storage cells.

# THE NEW YORK PUBLIC LIBRARY

# THE NEW YORK PUBLIC LIBRARY

## BRIEF DESCRIPTION OF THE DRAWINGS

[0007] **Figure 1** shows a memory unit.

[0008] **Figure 2a** shows an embodiment of a resetable memory that is constructed with a memory unit that does not have a reset function.

[0009] **Figure 2b** shows another embodiment of a resetable memory that is constructed with a memory unit that does not have a reset function.

[0010] **Figure 2c** shows an embodiment of a register transfer level (RTL) description that may be used to describe the resetable memory of **Figure 2a**.

[0011] **Figure 3** shows an embodiment of a methodology that can be executed by the resetable memory of Figure 2.

[0012] **Figure 4** shows a design methodology that automatically incorporates a resetable memory that is constructed with a memory that does not have a reset function into a semiconductor chip design.

[0013] **Figure 5** shows another embodiment of a resetable memory that is constructed with a memory unit that does not have a reset function.

[0014] **Figure 6** shows another embodiment of a resetable memory that is constructed with a memory unit that does not have a reset function.

[0015] **Figure 7** shows an embodiment of a methodology that may be used to automatically install a resetable memory having a reset value write unit as observed in Figure 6.

[0016] **Figure 8** shows an embodiment of a computing system.

## DETAILED DESCRIPTION

[0017] A resetable memory is described that includes a memory without reset capability having a data output coupled to a first input of a first multiplexer. A second input of the first multiplexer has a reset value input. A channel select for the first multiplexer is coupled to a resetable storage cell output that indicates whether a storage cell within the memory without reset capability has been written to after a reset or has not been written to after a reset. Other methods and apparatus are also described.

### Resetable Memory Unit Embodiment(s)

[0018] **Figure 2a** shows an embodiment of a resetable memory 220 that is constructed with a memory unit 201 that does not have a reset function. The memory unit without reset 201 of **Figure 2a** can be viewed as corresponding to the memory unit without reset 101 that was originally described in **Figure 1**. The resetable memory 220 of **Figure 2a** includes both a memory without reset 201 (as described above) as well as a memory unit 205 that includes a reset function (i.e., a memory unit 205 with reset).

[0019] In various embodiments, the data width of the memory unit with reset 205 is “smaller” than the data width of the memory unit without reset 201. That is, in various approaches, the cell word size of the memory unit with reset 205 is less than the cell word size of the memory cell without reset 201. In a further embodiment, the cell word size of the memory unit with reset 205 is one bit wide (i.e., such that its output line 210 carries only two states “1” or “0”).





**[0023]** The multiplexer 207 is configured to present at the memory unit data output 209: 1) the reset value 208 if the resetable memory unit 220 cell being addressed during a read operation has not been written to since its last “reset”; and 2) the data word read from the memory unit 201 (as provided on the actual memory unit data output 206) if the resetable memory unit 220 cell being addressed during the read operation has been written to since its last “reset”. In this manner, the memory unit output 209 presents data consistent with the operation of a memory unit having the ability to reset the data words that it stores.

**[0024]** That is, after the resetable memory 220 has been “reset”, any attempt to read a data word from a particular cell within the memory unit without reset 201 will produce the reset value 208 at the memory unit data output 209. This functional behavior continues for each cell until a particular cell is written to. After a particular cell is written to, the newly written data is presented at the memory unit data output 209 until the next reset of the resetable memory 220 occurs. The multiplexer 207 and memory unit with reset 205 may be viewed as providing for the emulation of a reset function for the memory unit without reset 201 reset.

**[0025]** In an embodiment where the data output 210 of the memory unit with reset 205 (which feeds the channel select input of multiplexer 207) is one bit wide, the data output 210 effectively acts a state bit that signifies whether the resetable memory 220 cell being read from has been: 1) not written to since a last resetable memory 220 reset; or 2) has been written to since a last resetable

memory 220 reset. Note that in the particular embodiment of **Figure 2a** a “0” is used to signify the former and a “1” is used to signify the later.

[0026] The memory unit with reset 205 acts a storage resource that keeps a record, for each cell within the memory unit without reset 201, as to whether or not the particular cell has been written to after a reset of the resetable memory 220. The reset input of the memory unit with reset 205 acts as the reset input 219 for the resetable memory 220. When the reset 219 is active, each cell within the memory unit with reset is cleared (e.g., is given a value of “0”). Noting that the Write Enable 204, Address 202, and Read Enable 230 lines of both memory units 201, 205 are tied together, any attempt to read a particular cell within the memory unit without reset 201 causes a simultaneous read from the memory unit with reset 205 at its corresponding cell.

[0027] Thus, if an attempt is made to read a cell after a reset (and before any writing to the cell has occurred), the reset output value of the memory unit with reset 205 (e.g., a “0”) will be presented to the multiplexer’s 207 channel select input. According to the embodiment of **Figure 2**, a channel select value of “0” causes the reset value 208 to be presented at the resetable memory output 209. As such, the resetable memory output 209 properly reflects a reset value for a cell that has not been written to after a reset has been applied.

[0028] Similarly, any attempt to write to a particular cell within the memory unit without reset 201 causes a simultaneous write into the memory unit with reset 205 at its corresponding cell. Noting that the data input 214 of the memory unit with reset 205 is tied to a “1” in the embodiment of **Figure 2**, the first writing

to a particular cell within the memory unit without reset 201 will cause the corresponding cell within the memory unit with reset 205 to have its reset value (of "0") to be written over (with a value of "1").

**[0029]** Thus, subsequently, if an attempt is made to read the same cell before a next a reset occurs, an output value of "1" will be presented from output 210 of the memory unit with reset 205 to the multiplexer's 207 channel select input.

According to the embodiment of **Figure 2**, a channel select value of "1" causes the actual memory unit without reset output 206 to be presented at the resetable memory output 209. As such, the resetable memory output 209 properly reflects the most recently written cell value for those cells that have been written to after a reset has been applied.

**[0030]** In this manner, the circuit of **Figure 2a** emulates the behavior of a memory unit having the storage capacity of memory unit 201 but also having reset capability. For each cell within the memory unit without reset 201, once the resetable memory 220 is reset, the reset value 208 should be observed for all subsequent read operations until the cell is written to. After the cell is written to, the value of the most recent written data should be observed until the next reset of the resetable memory 220 occurs.

**[0031]** **Figure 3** shows a methodology that reviews the approaches discussed above. First, a reset is asserted 301. This causes the reset value 208 to be provided 302 as the effective memory cell output - until the cell has been written to. If the cell has been written to, the actual value of the cell contents as stored within the memory unit without reset 201 are presented 303 as the output.

[0032] Before continuing note that the particular embodiment of **Figure 2a** indicates that the same reset value 208 is used for each resetable memory 220 cell. In alternate embodiments, different reset values may be used for various groupings of cells (or individual cells themselves). For example, referring to **Figure 2b** a reset value function circuit 230 may be inserted between (and coupled to) the reset value 208 input of the multiplexer 207 and the address input 202 of the resetable memory 220. The circuit 230 may be designed to issue different reset values, depending on the particular addressing value that appears on the address input 202.

#### Automatic Inference and Implementation Of Memory With Selective Reset

[0033] A further utility of the approaches discussed above is the ease at which a memory having reset may be incorporated into a designer's circuit design – even if the underlying semiconductor technology does not easily provide for a memory core having selective reset. For example, semiconductor circuits are typically designed with a particular semiconductor manufacturing process (i.e., a “foundry”) in mind. Usually, the foundry supplies models of basic building blocks (e.g., logic gates, memory units, etc.) from which a semiconductor chip design can be constructed.

[0034] The resetable memory approach discussed above may be used to effectively provide a designer with a resetable memory, even though the foundry only manufactures traditional memory units without reset (such as the traditional memory unit discussed with respect to **Figure 1**); or, offers resetable memories that are too cumbersome to implement as a large memory. For example, in one

**[0035]** In one embodiment, the use of a resetable memory is automatically inferred from the behavioral level or RTL level description of the designer's design; and, in response, a resetable memory is automatically inserted into the circuit designer's design. A behavioral level or RTL level description is a circuit description that is tailored to be understood by a computer and that describes the circuit in terms of its methodology (e.g., the various processes performed by the circuit and the relationship(s) between them) as opposed to describing the circuit only in terms of its hardware components and the interconnections between them (e.g., gates, registers, signal lines, etc.).

[0036] Given that behavioral or RTL level descriptions are written in terms of methodologies to be understood by a computer, they are frequently documented in a form that is similar to a software program or a pseudo code description of the operational flow of the circuit. As an example, **Figure 2c** provides an embodiment of an RTL level description that can be used to describe the resettable memory of **Figure 2a**. In order to automatically “install” a resettable memory into a circuit designer’s design, a software design tool can be configured to automatically infer the use a resettable memory from a behavioral level or RTL level description by identifying from the operational flow of the circuit description that a reset condition is being individually applied to one or more variables (e.g.,

that is to be implemented in hardware as a data word that resides within some type of storage cell).

**[0037]** Furthermore, in response to the inferred use of a resetable memory, the software tool may be further designed to automatically install (or present the designer with an offer to automatically install) a resetable memory into the designer's circuit. In one embodiment, the resetable memory 220 is incorporated into the designer's circuit at the gate level. Typically, an RTL level description of a circuit is compiled into a gate level netlist that describes the specific digital structures used in the design and the interconnections between them. Thus, upon recognition of a reset condition within the behavioral level or RTL level description of the circuit, the compilation result into its corresponding gate level form is modified by the design tool so as to incorporate the resetable memory.

**[0038]** In a further embodiment, if a memory unit with reset is offered by the applicable foundry, the foundry offering is incorporated into the resetable memory as the memory unit with reset 205. Further still, the design tool may be configured with its own design library that includes a memory with reset 205 that can be configured from a foundry's standard logic offerings. As result, the resetable memory 220 can be constructed even for those foundries that do not offer a memory with reset in their standard offerings. An embodiment of a gate level memory with reset that can be constructed from standard logic is provided in more detail ahead.

**[0039]** **Figure 4** shows a methodology that reviews such a design flow sequence. First, the use of a resetable memory is automatically inferred 401

from the designer's behavioral or RTL level description . The automatic inference can be accomplished, for example, by configuring the design tool to recognize from the operational flow of the circuit that: 1) some type of reset is being applied to the stored data values within the circuit; and 2) the stored data values are being changed to some type of reset value in response. After the use of a resettable memory has been inferred 401, a resettable memory is incorporated into the designer's design (e.g., at the gate level as described just above).

**[0040]** Note therefore that embodiments of the present approach (as well as those discussed further ahead) may be implemented not only within a semiconductor chip but also within machine-readable media. For example, the designs discussed above may be stored upon and/or embedded within machine readable media associated with a design tool used for designing semiconductor devices. Thus, it is also to be understood that embodiments of this invention may be used as or to support a software program executed upon some form of processing core (such as the CPU of a computer) or otherwise implemented or realized upon or within a machine-readable medium. A machine-readable medium includes any mechanism for storing or transmitting information in a form readable by a machine (e.g., a computer). For example, a machine readable medium includes read only memory (ROM); random access memory (RAM); magnetic disk storage media; optical storage media; flash memory devices; electrical, optical, acoustical or other form of propagated signals (e.g., carrier waves, infrared signals, digital signals, etc.); etc.

## Resettable Memory With Gate Level Design For Memory Unit With Reset

**[0041]** **Figure 5** relates to resettable memory design embodiment that can use standard foundry logic to create a memory unit with reset. As such, the design of **Figure 5** can be used even if the underlying foundry does not provide a memory with reset in its standard design library. According to the design embodiment of **Figure 5**, a resettable register 511 is used as a storage cell for the memory unit with reset. Here, the register 511 holds the value of a state bit that controls the channel select of the multiplexer 507 for its corresponding cell.

**[0042]** The register 511 also has a reset input 519 used to effectively reset the storage cell. When a reset signal is presented to the register 511, the state bit from the register output is set to a "0". As such, the channel select line 510 of the multiplexer 507 is positioned to enable the reset value input 508 of the multiplexer 507. According to the design approach of **Figure 5**, the register 511 output is coupled in a feedback arrangement to an input of a second multiplexer 513. The other input 514 of the second multiplexer 513 corresponds to a logical value that is opposite that of the register 511 reset value. That is, because the register 511 reset value is "0" in the embodiment of **Figure 2**, a "1" is provided to the input 514 of the second multiplexer 513.

**[0043]** Note that, after a reset, the feedback input to the second multiplexer 513 will be a "0" because the register 511 reset value is "0". The second multiplexer 513 is configured to transition from the enablement of the feedback input to the enablement of the "1" input 514 when the memory unit cell is first written to after a reset. That is, for as long as the memory unit cell is not written



to after a reset (i.e., is either reset again, read from or not used at all), the write enable (WE) line 504 and ADDR\_HIT line 517 remain at a logic "0".

**[0044]** As such, during this time period, the value of the channel select line 515 for the second multiplexer 513 is a logic "0". This corresponds to the register 511 being continuously latched with its reset output value of "0". As described above, this further corresponds to the channel select line 510 of the first multiplexer 507 being positioned to enable the reset value input 508 of the multiplexer 507. In the embodiment of **Figure 5**, the WE line 504 is a logic "1" only if a write operation occurs; and, the ADDR\_HIT line 517 is a logic "1" only if the address corresponding to the memory unit without reset 501 cell represented by register 511 appears at the address input 502.

**[0045]** As such, the output 515 of the AND gate 516 is a logic "1" only if the memory unit without reset 501 cell represented by register 511 is being written to. When this occurs, the "1" input 514 to the second multiplexer 513 is enabled, and a "1" is latched into the register 511. Thus, due to the feedback input of the second multiplexer, a "1" is continuously latched into the register 511 after the first write operation is performed upon the memory unit cell (after it has been a reset); and, a "0" does not appear at the register 511 output until it is next reset.

**[0046]** This corresponds to the actual memory unit data output 506 being enabled by the first multiplexer 507 after the memory unit cell is first written to (after it has been reset). The actual memory unit data output 506 is then enabled until the memory unit cell is reset again. Before continuing, note that the

ADDR\_HIT 517 may be provided by a logic circuit (not shown in **Figure 5** for simplicity) that effectively decodes the address input 502 value.

[0047] For simplicity only one register 511 is shown in **Figure 5**. However, it should be understood that each of the various inputs of multiplexer 550 may be coupled to its own register and corresponding circuit. That is, for example, the circuitry outlined by region 560 may be repeated at each multiplexer 550 input to provide reset capability for a plurality of cells. Note that the ADDR\_HIT input 517 associated with each repeated circuit should register a logical “1” only when the particular memory unit cell that the circuit applies to is accessed.

[0048] The third multiplexer 550 collects each register output. During a memory read from a memory unit cell having selective reset capability, the address of the memory unit cell to be read is used as an input to the third multiplexer 550 so that the reset state bit from the appropriate register (i.e., the register that corresponds to the memory unit cell being read from) is delivered to the first multiplexer 507. As such, the channel select 510b of the first multiplexer is controlled on a “per cell” basis.

#### First Alternate Embodiment of Resettable Memory

[0049] **Figure 6** shows another resettable memory embodiment 620. The resettable memory 620 of **Figure 6** includes a memory unit without reset 601 and a reset value write unit 640 and reset input 619 that are coupled to the write enable 604, address 602 and data input 603 of the memory unit without reset 601. According to one approach, in response to an active reset being applied at the reset input 619, the reset value write unit 640 writes a reset value into each

cell of the memory unit without reset 601. Thus, similar to the approaches described above, the memory unit without reset 601 “appears” to circuitry downstream from output 606 “as if” it is resettable.

[0050] Because an extended amount of time may be consumed by the reset value write unit 640 in the writing of a reset value into a number of memory unit without reset 601 cells, the resettable memory embodiment 620 of **Figure 6** is most appropriate in those environments where a write operation to the memory unit without reset 601 does not occur for an extended period of time after the reset input 619 is activated. As such the write enable input 604, the address input 602 and the data input 603 are sufficiently free to accept the input signals provided by the reset value write unit 640 (while reset values are being written into the memory unit without reset 601) without interruption from sources outside the resettable memory 620. The reset value write unit 640 may be designed, as just one embodiment, to repeatedly: 1) increment a prior address value so as to form a current address value; and, 2) write a reset value at the current address value.

[0051] Here, an appropriate environment for the type of resettable memory 620 observed in **Figure 6** can be determined from an analysis of the number of available clock cycles that may be used to execute a reset. As clock cycles can be inferred from within a behavioral or RTL level environment, the above described analysis can be performed at the behavioral level or RTL level as well. Generally, the more clock cycles that may be consumed in executing a reset, the more likely it is that the approach of **Figure 6** can be successfully implemented.

**[0052]** As such, a design tool may be configured to not only infer the use of a resettable memory from a behavioral or RTL level description, but may also be configured to infer the number available of clock cycles for executing a reset to one or more variables. This may be accomplished, for example, by estimating or calculating the time (e.g., by counting the number of clock cycles) between the moment a reset is issued to a particular variable; and, the earliest moment when the reset variable could be subsequently needed by the operational flow of the circuit.

**[0053]** Furthermore, a determination as to the amount of time that is available to implement a reset may be used to drive which particular type of resettable memory (e.g., a resettable memory 220 having a memory unit with reset 205 as observed in **Figure 2a**; or, a resettable memory 620 having a reset value write unit 640 as observed in **Figure 6**) is to be automatically installed by a software design tool. **Figure 7** shows an embodiment of a methodology that describes such an approach in more detail. Here, as in the methodology of **Figure 4**, the existence of a resettable memory is inferred 701 from the operational flow of a behavioral or RTL level circuit description.

**[0054]** Then, the number of available of clock cycles available for implementing a reset are inferred 702. Here, if the number of clock cycles is small enough so as to be less than a threshold value "X" (which, in turn, corresponds to the amount of time consumed by the reset value write unit 640) then the use of a resettable memory having a reset value write unit 640 is de-emphasized (because the reset value write unit 640 may need more time to

execute the reset than the circuit can wait to use the reset value). As such, a resettable memory having a memory unit with reset and a memory unit without (e.g., as observed in **Figure 2a**) is automatically installed 705. By contrast, if the number of clock cycles is large enough so as to be greater than the threshold value "X" then the use of a resettable memory having a reset value write unit 640 is emphasized (because the reset value write unit 640 has a sufficient amount of time to execute a reset). As such, a resettable memory having a reset value write unit 640 (e.g., as observed in **Figure 6**) is automatically installed 704. Here, often, the approach of **Figure 6** consumes less semiconductor chip surface area than the approach of **Figure 2a**; and, as such, in an effort to efficiently consume semiconductor surface area, the methodology of **Figure 7** installs the approach of **Figure 2a** only if the approach of **Figure 6** is deemed to be unfeasible.

[0055] **Figure 8** shows an embodiment of a computing system 800 that can execute instructions residing on a machine readable medium (noting that other (e.g., more elaborate) computing system embodiments are possible). The instructions may be related to integrated circuit design (e.g., as described in Figures 2a through 7). In one embodiment, the machine readable medium may be a fixed medium such as a hard disk drive 802. In other embodiments, the machine readable medium may be movable such as a CD ROM 803, a compact disc, a magnetic tape, etc. The instructions (or portions thereof) that are stored on the machine readable medium are loaded into memory (e.g., a Random Access Memory (RAM)) 805; and, the processing core 806 then executes the

